

1. Classes, Objets et le Diagramme de Classes UML

M2104 - Conception Objet

Édition 2021

Conception et modélisation

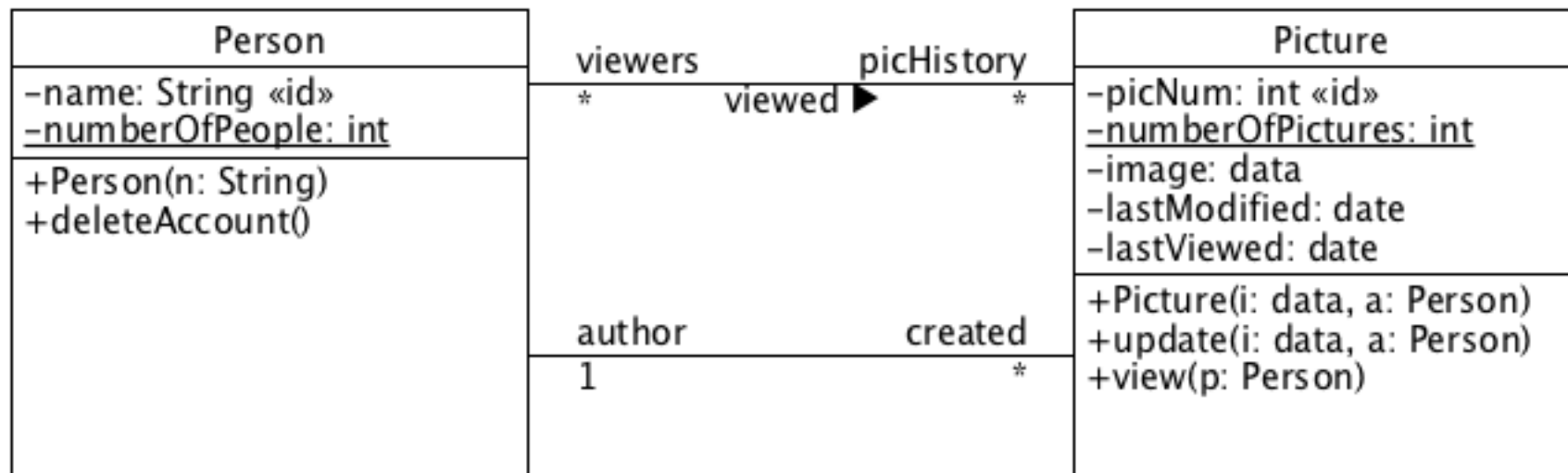
- ❖ **La conception** a pour but de préparer le développement logiciel, de clarifier les besoins du projet. La conception est **synthétique**, produit des **spécifications** logicielles, et est une activité **créative**.
- ❖ **La modélisation** est la fabrication d'un **modèle** d'un phénomène. C'est l'**abstraction** d'une partie du monde réel / imaginaire. La modélisation est **analytique**, produit des **spécifications** de processus, et est un **résumé**.
- ❖ Avec à la fois **conception et modélisation** dans le **paradigme objet**, les développeurs créent des **spécifications logicielles complètes** incluant **besoins et comportements**.
- ❖ Le but de ces documents de spécification est d'être une **référence** pour le développement et d'être **communiqués** entre équipes.

Classes et objets

- ❖ Les **objets** sont des **instances** de classes, les **classes** définissent les objets.
- ❖ Les objets **fournissent des services** et contiennent les **données** pertinentes.
- ❖ Les services fournis sont des **opérations**, les **méthodes**, les **données** nécessaires sont les **attributs**.
- ❖ Les classes peuvent aussi fournir des **services et données indépendants d'entités** sous la forme de **méthodes de classe** et d'**attributs de classes** (non liées à un quelconque objet).
- ❖ Les **associations entre classes** sont des références entre objets de ces classes (un objet de l'une contient une référence à un ou des objet(s) de l'autre).
- ❖ Les objets associés et ceux **utilisant** d'autres classes forment un diagramme de relations, le **diagramme de classes UML**.

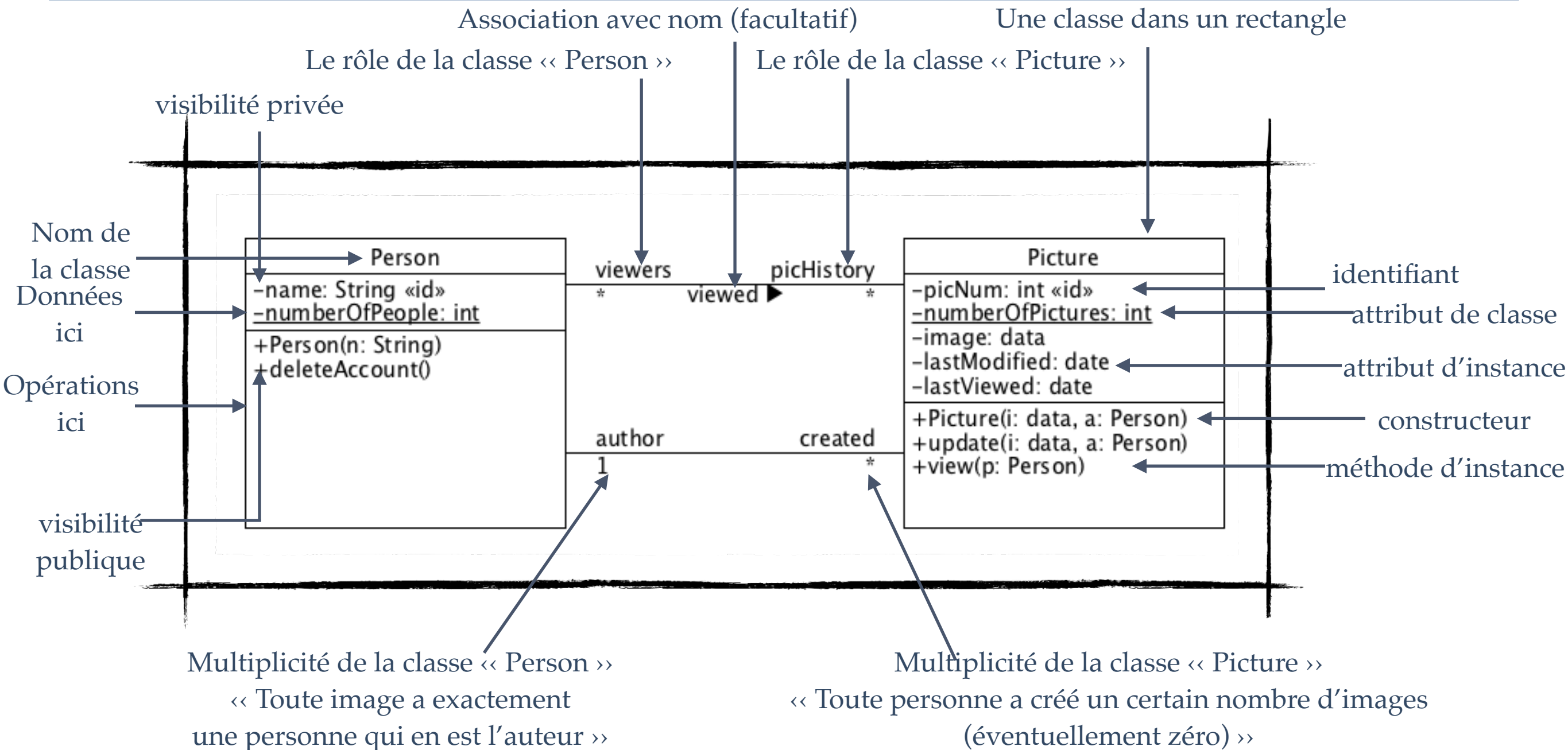
Le diagramme de classes UML

Exemple



Le diagramme de classes UML

Exemple



Le diagramme de classes UML

Récapitulatif

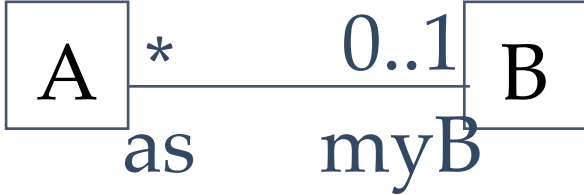

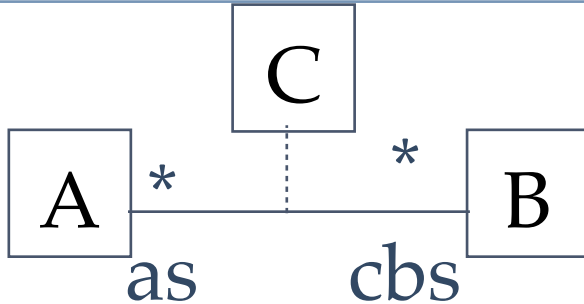
	UML	Sens	Java
Classe	Rectangle	Un « moule » pour objets	<code>class</code>
Attribut	visibilité nom : type	Donnée(s)	<code>visibilité type nom;</code>
Methode	visibilité nom(liste de paramètres) : type de retour	Opération	<code>visibilité type nom (liste de paramètres) {}</code>
Visibilité	+ / - / ~ / #	Fourni aux autres / que pour cette classe / ...	<code>public / private / / protected</code>
De classe / d'instance	Souligner pour classe	Les attributs et méthodes de classe ne sont pas liés à un objet en particulier	<code>static</code>
Associations	Trait continu avec rôles et multiplicités	Le fait que les objets fassent référence à d'autres objets	Attributs des classes respectives

Associations et conteneurs

- ❖ Multiplicité 1 ou 0..1 : une référence pouvant être nulle à un objet.
- ❖ Pluralités : conteneurs (incluant *collections*), comme :
 - ❖ Tableaux basiques java / listes (*ArrayList*)
 - ❖ Ensembles : *Set* (*TreeSet*, *HashSet*)
 - ❖ Tableaux associatifs : *Map* (*TreeMap*, *HashMap*)
- ❖ Les classes-association peuvent représenter des tableaux associatifs ou d'autres types de relations

Associations et conteneurs

Exemples

Multiplicité simple	 <pre>classDiagram class A class B A "*" -- "0..1" B : as, myB</pre>	<pre>class A { B myB; }</pre>
Multiplicité plurielle	 <pre>classDiagram class A class B A "*" -- "*" B : as, myBs</pre>	<pre>class A { List myBs; }</pre>
Classe-association	 <pre>classDiagram class A class B class C A "*" -- "*" B : as, cbs C A_B_association</pre>	<pre>class A { Map<B, C> cbs; }</pre>